

(11)Publication number : 10-063517

(43)Date of publication of application : 06.03.1998

(51)Int.Cl.

G06F 9/46

(21)Application number : 08-239774

(71)Applicant : NIPPON TELEGR & TELEPH CORP
<NTT>

(22)Date of filing : 22.08.1996

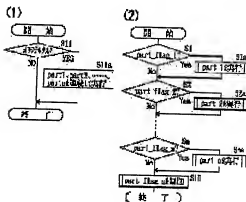
(72)Inventor : NAGANUMA JIRO
ENDO MAKOTO

(54) REAL TIME INFORMATION PROCESSING METHOD AND ITS DEVICE

(57)Abstract:

PROBLEM TO BE SOLVED: To provide real time information processing method and its device capable of excluding a hardware for controlling the priority of multi-interruption which is essential to a conventional example, a memory for a context switch necessary for switching an original task and the overhead of processing time, etc., in real time processing.

SOLUTION: At the time of processing plural original tasks on single CPU by time-multiplexing, each of the original tasks are divided to plural sub-tasks smaller than the original task of its own to select one of these divided plural subtasks to execute. Then in order that the number of the starting times of a basic polling loop which starts the subtasks once for each original task may be at least once within a prescribed and fixed time, the division is executed.



LEGAL STATUS

[Date of request for examination] 15.12.1998

[Date of sending the examiner's decision of rejection] 15.05.2001

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection] 2001-09961

[Date of requesting appeal against examiner's decision of rejection] 14.06.2001

[Date of extinction of right]

特開平10-63517

(43) 公開日 平成10年(1998) 3月6日

(51) Int.Cl.⁶

G 0 6 F 9/46

識別記号

3 4 0

庁内整理番号

F I

G 0 6 F 9/46

技術表示箇所

3 4 0 E

3 4 0 B

審査請求 未請求 請求項の数 6 F D (全 12 頁)

(21) 出願番号 特願平8-239774

(22) 出願日 平成8年(1996) 8月22日

(71) 出願人 000004226

日本電信電話株式会社

東京都新宿区西新宿三丁目19番2号

(72) 発明者 長沼 次郎

東京都新宿区西新宿三丁目19番2号 日本

電信電話株式会社内

(72) 発明者 遠藤 真

東京都新宿区西新宿三丁目19番2号 日本

電信電話株式会社内

(74) 代理人 弁理士 川久保 新一

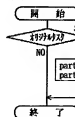
(54) 【発明の名称】 リアルタイム情報処理方法およびその装置

(57) 【要約】

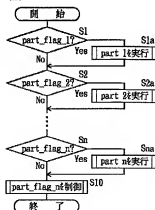
【課題】 リアルタイム情報処理において、従来例に不可欠な多重割り込みのプライオリティ制御用のハードウェアと、オリジナルタスク切替えに必要なコンテキストスイッチ用のメモリと、処理時間と等のオーバーヘッドを排除することができるリアルタイム情報処理方法および装置を提供することを目的とするものである。

【解決手段】 複数のオリジナルタスクを、1つのCPU上で時間多重で処理する場合、上記各オリジナルタスクを、自らのオリジナルタスクよりも小さい複数のサブタスクに分割し、この分割された複数のサブタスクからその1つを選択し、実行するものであり、各オリジナルタスク毎に上記サブタスクを1回づつ起動する基本ボーリングループの起動回数が、所定の一定時間以内に少なくとも1回になるように、上記分割が行われるものである。

(1)



(2)



【特許請求の範囲】

【請求項1】 複数のオリジナルタスクを、1つのCPU上で時間多重で処理するリアルタイム情報処理方法において、

上記各オリジナルタスクを、自らのオリジナルタスクよりも小さい複数のサブタスクに分割するオリジナルタスク分割段階と；上記分割された複数のサブタスクからその1つを選択し、実行するサブタスク選択・実行段階と；を有し、上記各オリジナルタスク毎に上記サブタスクを1回づつ起動する基本ポーリングループの起動回数10 が、所定の一定時間以内に少なくとも1回になるように、上記分割が行われ、所定のリアルタイム情報処理を実現することを特徴とするリアルタイム情報処理方法。

【請求項2】 1つのCPUと専用ハードウェアとで構成され、上記CPUと上記専用ハードウェアとの間で情報の授受を行いながら処理を進める情報処理装置によって、複数のオリジナルタスクを、上記CPU上で時間多重で処理するリアルタイム情報処理方法において、上記各オリジナルタスクを、自らのオリジナルタスクよりも小さい複数のサブタスクに分割するオリジナルタスク分割段階と；上記分割された複数のサブタスクからその1つを選択し、実行するサブタスク選択・実行段階と；を有し、上記各オリジナルタスク毎に上記サブタスクを1回づつ起動する基本ポーリングループの起動回数20 が、所定の一定時間以内に少なくとも1回になるように、上記分割が行われ、所定のリアルタイム情報処理を実現することを特徴とするリアルタイム情報処理方法。

【請求項3】 複数のオリジナルタスクを、1つのCPU上で時間多重で処理するリアルタイム情報処理装置において、

上記各オリジナルタスクを、自らのオリジナルタスクよりも小さい複数のサブタスクに分割するオリジナルタスク分割手段と；上記分割された複数のサブタスクからその1つを選択し、実行するサブタスク選択・実行手段と；を有し、上記各オリジナルタスク毎に上記サブタスクを1回づつ起動する基本ポーリングループの起動回数30 が、所定の一定時間以内に少なくとも1回になるように、上記分割が行われ、所定のリアルタイム情報処理を実現することを特徴とするリアルタイム情報処理装置。

【請求項4】 1つのCPUと専用ハードウェアとで構成され、上記CPUと上記専用ハードウェアとの間で情報の授受を行いながら処理を進める情報処理装置によって、複数のオリジナルタスクを、上記CPU上で時間多重で処理するリアルタイム情報処理装置において、上記各オリジナルタスクを、自らのオリジナルタスクよりも小さい複数のサブタスクに分割するオリジナルタスク分割手段と；上記分割された複数のサブタスクからその1つを選択し、実行するサブタスク選択・実行手段と；を有し、上記各オリジナルタスク毎に上記サブタスクを1回づつ起動する基本ポーリングループの起動回数40

が、所定の一定時間以内に少なくとも1回になるように、上記分割が行われ、所定のリアルタイム情報処理を実現することを特徴とするリアルタイム情報処理装置。

【請求項5】 請求項3または請求項4において、上記リアルタイム情報処理装置は、1つのチップ上に構成され、所定のリアルタイム情報処理を実現することを特徴とするリアルタイム情報処理装置。

【請求項6】 請求項3～請求項5のいずれか1項において、

上記所定のリアルタイム情報処理は、複数の入力ポートから入力された複数の入力ストリーム中のデータを、所定のシンタックスに従って、データの組立、分解、挿入、削除、加工、並び替え等の処理を行い、複数の出力ポートへ複数の出力ストリームとして出力するプロセッサ処理であることを特徴とするリアルタイム情報処理装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、リアルタイム情報処理において、従来不可欠であった多重割り込みのブラオリティ制御用のハードウェアと、オリジナルタスク切替えに必要なコンテキストスイッチ用のメモリと、処理時間と等のオーバヘッドを排除するリアルタイム情報処理に関するものである。

【0002】また、本発明は、CPUと専用ハードウェアとの間で情報の授受を行いながら処理を進める情報処理装置（以下、「エンベディッドシステム」と呼ぶ）上に、または、それらを1つのチップ上に集積したオンチップ・エンベディッドシステム(On Chip Embedded System)上に構築するリアルタイム情報処理に関するものである。

【0003】

【従来の技術】従来、複数のオリジナルタスク（処理の主体）を、1つのCPU（計算機）上で時間多重で処理を行い、しかも、所定の一定時間以内に全てのオリジナルタスクの起動（呼び出し）を保證するリアルタイム情報処理方法として、下記文献[1]が示す方法を探るのが一般的である。また、1つのCPUと専用ハードウェアとで構成され、上記CPUと上記専用ハードウェアとの間で情報の授受を行いながら処理を進める情報処理装置において、上記CPU上で、複数のオリジナルタスクを時間多重で処理するリアルタイム情報処理方法としても、下記文献[1]が示す方法を探るのが一般的である。

【0004】なお、通常の計算機システムの構成として、文献[2]が示す構成が知られ、エンベディッドシステムの構成として、文献[3]が示す構成が知られている。

・文献[1]：J.L.Perterson 他、「オペレーティングシ

システムの概念", 培風館, 1987.

・文献[2]: J.L.Hennessy and D.A.Patterson, "Computer Architecture: A Quantitative Approach", Morgan Kaufmann Publishers, Inc., 1990.

・文献[3]: Daniel D.Gaiski 他, "Specification and Design of Embedded Systems", Prentice Hall, 1994. 従来のリアルタイム情報処理方法における基本的な考え方は、上記文献[1]に示されているように、オリジナルタスクの緊急度に応じて各オリジナルタスクに優先度を付与し、その優先度順に、次に実行するオリジナルタスクを起動するものである。

【0005】この場合、現在実行中のオリジナルタスクよりも優先度の高いオリジナルタスクが起動されたときに「割り込み」が発生し、現在実行中のオリジナルタスクの処理を一旦中断し、上記起動された優先度の高いオリジナルタスクを先に実行し、この起動された優先度の高いオリジナルタスクの実行が終了した後に、上記中断していたオリジナルタスクの実行を再開する。すなわち、従来の方法は、割り込みを前提とする処理方法であり、この割り込み処理は、リアルタイム性を確保するためには不可欠な処理である。

【0006】

【発明が解決しようとする課題】ところで、上記文献[1]に示す従来のリアルタイム情報処理方法を、たとえば、上記文献[2]に示す通常の計算機システム、または、特に、文献[3]に示すエンベディッドシステム上で実現する場合、次の新たな問題が生じる。

【0007】上記エンベディッドシステム(Embedded System)は、CPUと専用ハードウェアとで構成され、CPU上のソフトウェアと専用ハードウェアとの間で情報の授受を行いながら処理を進めるシステムであり、このようにすることによって、専用ハードウェアの「高速性」とCPU上のソフトウェアによる「柔軟性」とを兼ね備えたシステムとして注目されているシステムである。

【0008】上記エンベディッドシステムにおいて、専用ハードウェア(HW)とソフトウェア(SW)との間で行われる情報の授受は、一般に、数十個を越えるHW/SWインタフェースレジスタを介して行われ、これらの情報の授受は、基本的に頻繁に発生ししかも非同期的に発生する。したがって、割り込みを前提とする従来のリアルタイム情報処理においては、複数のHW/SWインタフェースレジスタからの多重でしかも非同期的な割り込みをサポートしなければならないので、多重割り込みのプライオリティ制御を行うためには、ハードウェア規模が大きくなるという問題があり、しかも、ハードウェア全体の構成が複雑になるという問題がある。

【0009】また、優先度が付与された各オリジナルタスクの起動、中断、再開、優先度制御等を処理するために、リアルタイムOSをサポートしなければならないので、頻繁に発生するオリジナルタスク切替に必要なコ

ンテキストスイッチ用に、膨大なメモリと処理時間とが必要であるというソフトウェアの問題がある。たとえば、次の文献[4]、[5]に示すように、最も小さく軽いリアルタイムOSでも、数十kB〜数百kBのメモリを必要とする。

・[4] M.Accetta 他, "Mach: A New Kernel Foundation for UNIX Development", Proc. Summer 1986 USENIX, p. 93-112, 1986.

・[5] J.K.Ousterhout 他, "The Sprite Network Operating System", IEEE Computer, Vol. 21, No. 2, pp. 23-26, 1988.

上記ハードウェアの問題とソフトウェアの問題、換言すれば、要求条件は、エンベディッドシステム(通常はオンボード)としては、効率的ではないものの、現状では実現がまだ可能である。しかし、将来的なオンチップのエンベディッドシステムとしては、効率的ではないばかりでなく、非現実的なものとなることが想定される。

【0010】一方、上記割り込みを用いなくて、「MP EG2システムパートのプロトコル処理」を実現しようとすると、本来実現しなければならないリアルタイム処理が満足されないという問題が発生する。なお、「MP EG2システムパート」に関しては、下記文献[6]に規定されている。

・[6] Systems-Generic Coding of Moving Pictures and Associated Audio-ISO/IEC 13818-1 International Standard, November 11, 1994.

上記MPEG2システムでは、映像(Video)、音声(Audio)、ユーザ(User)等のデータを、MPEG2システム標準の伝送(TS)/音積(PS)メディアヘリアルタイムで多重(MUX)/分離(DMUX)する必要がある。多重化処理では、映像/音声/ユーザのエレメンタリーストリーム(ES)について、それぞれ専用のバケット(PES)化を図り、それらをトランスポートストリーム(TS)等に多重化する。この際、時刻管理のためのPCR、環境情報のPSI、情報を含まないNullバケット等も多重化する。

【0011】図9は、従来のMPEG2システムにおけるトランスポートストリームTSの多重化処理の動作を示すフローチャートである。

【0012】図9において、基本ボーリンググループ内に含まれる各オリジナルタスク(ボックス)は、リアルタイム処理され、各オリジナルタスクが作成する各バケットの多重化の割合は、各エレメンタリーストリームESの規定レートを保つ必要がある。たとえば、Video: 5.0Mbps, Audio: 256kbps, User: 64kbps, PCR: 100msecに1個、PSI: 1secに1個、全体のTSのレートは6.144Mbpsを保つ必要がある。

【0013】上記基本ボーリンググループ内で繰り返し起動される各オリジナルタスクの実行時間(実行サイク

ル)は、バッファ量やヘッダ検索結果に依存し、数サイクル〜数百サイクルと異なる。

【0014】一方、「トランスポートストリームTSの1パケット(1TS)」が、一定時間以内に処理しなければならない単位であるとする、各オリジナルタスクをリアルタイム処理するためには、トランスポートストリームTSの1パケットを処理する間に、各オリジナルタスクをそれぞれ少なくとも1回起動する必要がある。しかし、このように各オリジナルタスクを単純に起動しようとする、図10に示すように、各オリジナルタスクを1回ずつ起動する基本ボーリンググループを、1つのトランスポートストリームTSの時間(一定時間)内に少なくとも1回起動することが不可能である場合(基本ボーリンググループを起動すると、一定時間を越える場合)があり、この場合には、基本ボーリンググループを、一定時間内に少なくとも1回起動することを保証できないという問題がある。

【0015】エンベディッドシステム上で、従来のリアルタイム情報処理を実現するときに生じる上記ハードウェア、ソフトウェアの問題は、今後開発される種々のエンベディッドシステムにおいても同様に生じる問題であり、「高速度」と「柔軟性」とを両立させたエンベディッドシステムにおける「オンボード」から「オンチップ」への発展を考慮すると、今後ますます深刻な問題となる。

【0016】ところが、上記従来例は、上記文献【2】、【3】に示す計算機システム上に、文献【1】に示す割り込みを前提としたリアルタイム情報処理方法を実現するに留まり、現状では、上記問題すら指摘されておらず、勿論これらを解決する手法も提案されていない。

【0017】本発明は、リアルタイム情報処理において、従来例に不可欠な多重割り込みのプライオリティ制御用のハードウェアと、オリジナルタスク切替えに必要なコンテキストスイッチ用のメモリと、処理時間と等のオーバーヘッドを排除することができるリアルタイム情報処理方法およびその装置を提供することを目的とするものである。

【0018】また、本発明は、CPUと専用ハードウェアとで構成されるエンベディッドシステム(Embedded System)上に、または、それらを1つのチップ上に集積したオンチップ・エンベディッドシステム(On Chip Embedded System)上に、リアルタイム情報処理を構築することができるリアルタイム情報処理方法およびその装置を提供することを目的とするものである。

【0019】

【課題を解決するための手段】本発明は、複数のオリジナルタスクを、1つのCPU上で時間多重で処理する場合、上記各オリジナルタスクを、自らのオリジナルタスクよりも小さい複数のサブタスクに分割し、この分割された複数のサブタスクからその1つを選択し、実行する

ものであり、各オリジナルタスク毎に上記サブタスクを1回ずつ起動する基本ボーリンググループの起動回数が、所定の一定時間以内に少なくとも1回になるように、上記分割が行われるものである。

【0020】

【発明の実施の形態および実施例】図1は、本発明の一実施例であるリアルタイム情報処理装置RIDI1を示す図である。

【0021】リアルタイム情報処理装置RIDI1は、CPU(計算機)10と、専用ハードウェア20とで構成され、CPU10と専用ハードウェア20との間で情報の授受を行いながら、複数のオリジナルタスクを、CPU10上で時間多重でリアルタイムに処理する装置である。

【0022】専用ハードウェア20は、バッファメモリ1と、ライト制御部2と、リード制御部3と、データメモリ4と、複数(m個)の入力ポート7と、複数(n個)の出力ポート8とを有する。

【0023】バッファメモリ1は、2ポートメモリで構成され、入出力ストリームの一部を格納するバッファメモリである。ライト制御部2は、複数の入力ストリームをバッファメモリ1の第1のポートへ書き込みする動作を制御するライト制御部であり、リード制御部3は、バッファメモリ1の第2のポートから複数の出力ストリームへ読み出しする動作を制御するものである。データメモリ4は、2ポートメモリで構成され、シタックス処理のために作業データを一時的に格納するメモリである。

【0024】CPU10は、バッファメモリ1内のデータを参照し、読み出し、データメモリを用いてシタックス処理を実現し、シタックス処理されたデータをバッファメモリ1に書き込む処理を実現するプログラム可能な制御部である。

【0025】また、CPU10は、オリジナルタスク分割手段、サブタスク選択・実行手段の例である。オリジナルタスク分割手段は、上記各オリジナルタスク(処理の主体)を、自らのオリジナルタスクよりも小さい複数のサブタスクに分割する手段である。サブタスク選択・実行手段は、上記分割された複数のサブタスクからその1つを選択し、実行する手段である。なお、上記各オリジナルタスク毎に上記サブタスクを1回ずつ起動する基本ボーリンググループの起動回数が、所定の一定時間以内に少なくとも1回になるように、上記分割が行われる。

【0026】オリジナルタスク分割手段と、サブタスク選択・実行手段とによって、基本ボーリンググループを1回起動する場合の最大実行サイクルを抑制する。

【0027】リアルタイム情報処理装置RIDI1は、メモリ分散化による並列処理を導入している。つまり、バッファメモリ1とデータメモリ4とを分離することによって、ストリーム入出力処理とシタックス処理とを並

列化している。また、バッファメモリ1とデータメモリ4とを2ポート化することによって、ハードウェアとソフトウェアとからのアクセスを並列化し、ストリームの入力処理と出力処理とを並列化し、これによって、入力側から出力側への不要な転送を排除する装置である。

【0028】図1に示すリアルタイム情報処理装置R1D1において、その横線の破線の上と下とは互いに独立に動作する。つまり、バッファメモリ1とライト制御部2とリード制御部3とで構成される組と、データメモリ4とCPU10とで構成される組とは、互いに独立的に動作する。

【0029】図2は、上記実施例におけるMPEG2準拠のシステムパートが規定するプロトコル処理の概要を示す図である。

【0030】比較のために、ソフトウェアによる実現方式の場合の動作概要も示してある。ソフトウェアで実現した場合、図中の各処理がソフトウェアで処理されるために長時間を要し、さらに、その各処理が逐次的に動作するため、膨大な時間が必要となる。ところが、上記実施例では、各処理部の専用ハードウェア化とそれらの並列パイプライン動作とによって、極めて高速に動作することができ。

【0031】リード制御部3、ライト制御部2が専用ハードウェア化されていることによって、全体を専用ハードウェア化したときにおける高速の入出力制御と同じ高速性を実現している。また、2ポート化されたバッファメモリ1を用いることによって、ソフトウェアによるシンタックス処理と、専用ハードウェアによる入出力の処理とを並列して動作させることができる。また2ポート化されたデータメモリ4を用いることによって、バッファメモリ1からのデータメモリ1への転送、逆に、データメモリ4からバッファメモリ1へのデータ転送等も並列化される。

【0032】さらに、入出力ストリーム用のバッファメモリを共通化することによって、バッファメモリ1内の入力ストリームをバッファメモリ1内の出力ストリームへ転送する動作を不要化している。たとえば、MPEG2システムパートの多重化(MUX)処理においては、入力ストリームの殆どは、ヘッダ類等を付加し、出力ストリームに含まれている。すなわち、入出力バッファを分離すると、この間の不要な転送が頻発する。これらの一連の動作によって、全体として、図2に示したような並列パイプライン処理が可能になる。全体をソフトウェアで処理する方式と比較して、上記実施例では、数十倍以上の高速化を達成できる。

【0033】次に、MPEG2システムパートの多重化処理を例にとって、上記実施例の全体の動作概要をより詳細に説明する。この場合、入力ストリームとして、映像、音声、ユーザの3つのストリーム(エレメンタリストリーム)を例にとり、出力ストリームとして、トラン

スポートの場合を例にとる。

【0034】ライト制御部2(専用ハードウェア)は、入力ポート7から、映像、音声、ユーザデータを入力し、この入力した各データを、バッファメモリ1の異なるアドレス空間上に格納する。この場合、各データのバッファ内アドレス(開始と終了のアドレス)と、所定のヘッダとを検索し、その存在アドレスをCPU10に通知する。

【0035】CPU10は、ライト制御部2からの通知と上記エレメンタリストリームの参照とに従って、各エレメンタリストリームの所定のヘッダ類を作成し、バッファメモリ1の所定のアドレス空間に書き込み、ヘッダ類の長さをリード制御部3に通知する。なお、上記第2の実施例においては、CPU10は、実際には、そのCPU+ソフトウェア(プログラム)によって、シンタックス処理制御が実行される。

【0036】リード制御部3(専用ハードウェア)は、CPU10からの通知に従って、バッファメモリ1内のヘッダ類と所定のバケット長さからヘッダ類の長さを差引いた分の所定のエレメンタリストリームとを、出力ポート8に送出する。バッファメモリ1は、ライト制御部2、リード制御部3、CPU10の動作に従って、入出力ストリームの一部を格納する。データメモリ4は、CPU10における一時的な作業データを格納する。

【0037】上記各処理部は、上記動作を繰り返し実行し、MPEG2システムパートが規定する入力ストリームの多重化処理を実現し、出力ストリームを送出する。

【0038】図3は、上記実施例において、MPEG2システムパートが規定する2つの処理(通信メディア用のトランスポートストリーム処理、蓄積メディア用のプログラムストリーム処理)を実現するとき使用するバッファメモリ1の使用例について示す図である。

【0039】トランスポートストリーム処理とプログラムストリーム処理との違いは、MPEG2システムパートに規定されているが、それらのシンタックスは、それぞれの使用目的(通信/蓄積)に応じて異なっている。ここで重要なことは、入出力ストリームの出力タイミングが同一であれば、バッファメモリ1の使用方法(メモリマップ)を変更するだけ、同一のハードウェアによって、異なるシンタックスのプロトコル処理を実現できることである。すなわち、リアルタイム情報処理装置R1D1は、シンタックスの変更にも容易に対処可能な柔軟性を有している。

【0040】上記のように、リアルタイム情報処理装置R1D1は、図2に示す動作の高速性と、図3に示す柔軟性とが可能になる。つまり、リアルタイム情報処理装置R1D1は、所定のプロトコルに要求される高速性と、所定のプロトコルの変更等へ容易に対処可能な柔軟性とを兼ね備える。すなわち、専用ハードウェアによる高速性と、ソフトウェアによる柔軟性とを両立すること

ができる。これによって、MPEG2システムパートのような複雑で、まだ流動的で、また、リアルタイム処理が不可欠な高速度なプロトコルを容易に実現することができる。

【0041】次に、上記実施例において、従来必要な多重割り込みのプライオリティ制御用のハードウェアと、オリジナルタスク切替に必要なコンテキストスイッチ用のメモリと、処理時間と等のオーバーヘッドを排除することができる点について説明する。

【0042】次に、上記実施例において、オリジナルタスクT₀を複数のサブタスクTsに分割し、1つのサブタスクTsを選択し、実行する動作について説明する。

【0043】図4(2)は、上記実施例において、オリジナルタスクT₀を複数のサブタスクTsに分割し、その1つのサブタスクTsを選択し、実行する動作を示すフローチャートである。

【0044】図5は、タスクを実行するイメージを示す図であり、図5(1)は、従来の実行イメージを示す図であり、オリジナルタスクT₀が分割されていないので、図4(1)に示すように、当該オリジナルタスクであれば(S11)、part1、part2、……、partnが連続して実行される(S11a)。図5

(2)は、上記実施例において、サブタスクTs毎に実行するイメージを示す図であり、n個のサブタスクTsに分割され、part1、part2、……、partnのそれぞれが独立して実行される。

【0045】図4(2)において、まず、現在起動すべきサブタスクTsがpart1(サブタスクTsのうちで最初のサブタスクであれば(S1)、そのpart1を実行し(S1a)、現在起動すべきサブタスクTsがpart2(サブタスクTsのうちで2番目のサブタスクであれば(S2)、そのpart2を実行し(S2a)、……、現在起動すべきサブタスクTsがpartn(サブタスクTsのうちでn番目のサブタスク)であれば(Sn)、そのpartnを実行する(Sna)。そして、次に実行すべきサブタスクTsを設定する制御を行う(S10)。上記実施例においては、分割動作と選択動作とが同時に行われているが、分割動作と選択動作とを時間を遡って実行するようにしてもよい。

【0046】なお、図4において、「part_flag_n」は、オリジナルタスクT₀から分割された複数のサブタスクTsのうちでn番目のサブタスクTsであることを示す変数である。つまり、オリジナルタスクT₀は、変数「part_flag_n」で示されるn個のサブタスクTsに分割され、「part_flag_n」の示す部分だけがC言語の「if_then」コンストラクトで選択され、実行される。

【0047】ここで、変数「part_flag_n」の値は、排他的に制御される。すなわち、オリジナルタスクT₀が1回起動される毎に、分割されたn個のサブ

タスクTsのうち1個のサブタスクTsのみが選択され、実行されるように制御される。所定のオリジナルタスクT₀が起動される毎に、上記所定のオリジナルタスクT₀から分割された複数のサブタスクTsのうちで、異なるサブタスクTsが順次、選択される。たとえば、1番目のサブタスクTs(Part1)→2番目のサブタスクTs(Part2)→……→n番目のサブタスクTs(Partn)→1番目のサブタスクTs(Part1)→2番目のサブタスクTs(Part2)→……の順で選択される。なお、各サブタスクTs間で共有される変数はグローバルに扱う必要がある。つまり、各サブタスクTs間で共有すべき変数は、実行すべき対象のサブタスクTsが変わっても、常に使用できるようにしてあることが必要である。

【0048】上記一連の動作(S1～S10)を最初のオリジナルタスクT₀(たとえばVideo)について実行し、これが終了したら、次のオリジナルタスクT₀(たとえばAudio)について実行し、……、最後のオリジナルタスクT₀(たとえばPS1)について実行し、この一連の動作(基本ボーリンググループ)を繰り返す。

【0049】上記実施例において、各サブタスクTs(Part1, Part2, …, Partn)は、オリジナルタスクT₀を任意の場所まで分割されたものであり、オリジナルタスクT₀から分割されたサブタスクTsは、それを実行する場合、各サブタスクTsの個別の「call」から「return」まで実行され、サブタスクTsの実行サイクルがオリジナルタスクT₀の実行サイクルよりも短い。すなわち、オリジナルタスクT₀を分割した後におけるオリジナルタスクTsの実行サイクルは、オリジナルタスクT₀の実行サイクルよりも短い。しかも、所定のオリジナルタスクT₀が分割されたサブタスクTsの実行サイクルは、その分割の仕方によって、任意に設定することができ、したがって、1つの基本ボーリンググループ(各オリジナルタスク毎にサブタスクを1回ずつ起動するグループ)においては上記所定のオリジナルタスクT₀が部分的にしか実行されないが、上記所定のオリジナルタスクT₀の実行サイクルを実質的に任意の大きさに合わせることができ。なお、オリジナルタスクT₀が任意の大きさに分割され、実行されても、細切れではあるが、一連の処理が実行され、オリジナルタスクの実行という点では問題が生じない。

【0050】図6は、上記実施例における基本ボーリンググループ起動回数設定の説明図である。

【0051】まず、各オリジナルタスク毎にサブタスクTsを1回ずつ起動する基本ボーリンググループの起動回数N_{int}を、一定時間T_{max}以内に所定の一定回数以上に設定する。上記実施例における一定時間T_{max}は、上記実施例をMPEG2システムに適用した場合において、リアルタイム処理しなければならない一定時間

て、上記実施例を利用すると、オンチップ・エンベディッドシステムを容易に実現できるので、1チップのVLSI化が可能であり、従来のリアルタイム処理を実現するMPEG2のシステムパートのプロトコル処理装置に比べて、小型であり経済的であるリアルタイム情報処理可能なMPEG2システムパートのプロトコル処理装置を実現することができる。

【0064】また、上記実施例では、リアルタイム情報処理として、それを実現するためのハードウェアへの無理な要求がなく、極めて簡単な情報処理システムを提供することができる。VLSI上にインプリメントする場合、コプロセッサやメモリ等はライブラリとして提供可能なので、上記実施例を実現する場合、アプリケーションに必要な機能のみを専用ハードウェアで実現するだけで足りる。これらのアプリケーションに必要な機能は、近年の論理合成技術で極めて容易に実現できる。このように、上記実施例は、ライブラリ化された(部品化された)コンポーネントをベースとした極めて簡単な構成であるので、VLSI設計技術における上位合成技術(たとえばシステムレベルシミュレーション等)の対象のターゲットアーキテクチャとしても極めて有効であり、上位合成技術の恩恵を受け、近い将来、上記実施例をベースにしたリアルタイム情報処理システムを上位仕様記述に依りて、VLSI上に自動合成することが可能になる。

【0065】上記のように、将来的なネットワーク技術をベースにしたマルチメディア時代において、上記実施例によって提供される小型、経済化、VLSI化が可能な新しいリアルタイム情報処理装置の果たす役割は図り知れない。したがって、上記実施例は、CPUと専用ハードウェアとで構成されるエンベディッドシステム(Embedded System)上に、または、それらを1つのチップ上に集積したオンチップ・エンベディッドシステム(On Chip Embedded System)上に、リアルタイム情報処理を構築するに好適である。

【0066】なお、上記実施例において、オリジナルタスクT₀をサブタスクT_sへ分割する方法、その分割されたサブタスクT_sを選択的に実行させるための変数の使用の方法等は、任意に定めることができる。また、ターゲットのシステムの構成、通常の計算機システムまたはエンベディッドシステム、オンボードまたはオンチップ等、任意に定めることができる。さらに、上記実施例において、MPEG2システムパートのプロトコル処理を用いたが、その応用の種類等も、任意に定めることができる。また、上記実施例において、専用ハードウェア20の代わりに他の専用ハードウェアを使用するようにしてもよい。

【0067】なお、上記実施例を、1つのチップ上に構成し、所定のリアルタイム情報処理を実現するようにしてもよい。

【0068】さらに、上記実施例を方法の発明として把

握することができ、つまり、上記実施例は、1つのCPUと専用ハードウェアとで構成され、上記CPUと上記専用ハードウェアとの間で情報の授受を行いながら処理を進める情報処理装置によって、複数のオリジナルタスクを、上記CPU上で時間多重で処理するリアルタイム情報処理方法において、上記各オリジナルタスクを、自らのオリジナルタスクよりも小さい複数のサブタスクに分割するオリジナルタスク分割段階と、上記分割された複数のサブタスクからその1つを選択し、実行するサブタスク選択・実行段階とを有し、上記各オリジナルタスク毎に上記サブタスクを1回づつ起動する基本ボーリングループの起動回数が、所定の一定時間以内に少なくとも1回になるように、上記分割が行われ、所定のリアルタイム情報処理を実現することを特徴とするリアルタイム情報処理方法として把握することができる。

【0069】また、上記実施例を専用ハードウェアを使用しない方法に適用することができる。つまり、複数のオリジナルタスクを1つのCPU上で時間多重で処理するリアルタイム情報処理方法において、上記各オリジナルタスクを、自らのオリジナルタスクよりも小さい複数のサブタスクに分割するオリジナルタスク分割段階と、上記分割された複数のサブタスクからその1つを選択し、実行するサブタスク選択・実行段階とを有し、上記各オリジナルタスク毎に上記サブタスクを1回づつ起動する基本ボーリングループの起動回数が、所定の一定時間以内に少なくとも1回になるように、上記分割が行われ、所定のリアルタイム情報処理を実現することを特徴とするリアルタイム情報処理方法として把握することができる。

【0070】また、上記実施例を、専用ハードウェアを使用しない装置、つまり、複数のオリジナルタスクを、1つのCPU上で時間多重で処理するリアルタイム情報処理装置に適用することができる。

【0071】【発明の効果】本発明によれば、リアルタイム情報処理において、従来、不可欠な多重割り込みのプライオリティ制御用のハードウェアと、オリジナルタスク切替えに必要なコンテキストスイッチ用のメモリと処理時間と等のオーバーヘッドを排除することができるという効果を奏する。

【図面の簡単な説明】

【図1】本発明の一実施例のリアルタイム情報処理装置R1D1を示す図である。

【図2】上記実施例において、MPEG2準拠のシステムパートが規定するプロトコル処理の概要を示す図である。

【図3】上記実施例において、MPEG2システムパートが規定する2つの処理を実現するとき使用するバッファメモリ1の使用例について示す図である。

【図4】オリジナルタスクT₀またはサブタスクT_sを

実行する動作を示すフローチャートである。

【図5】タスクを実行するイメージを示す図である。

【図6】上記実施例における基本ボーリングループ起動回数設定の説明図である。

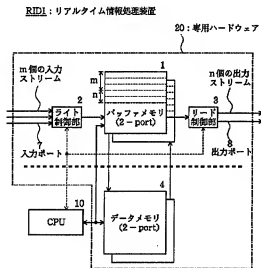
【図7】上記実施例をMPEG2システムに適用した場合の一例の説明図であり、基本ボーリングループBPLを起動する動作の概要を示す図である。

【図8】上記実施例をMPEG2システムに適用した場合の具体例を示す図である。

【図9】従来のMPEG2システムにおけるトランスポートストリームTSの多重化処理の動作を示すフローチャートである。

*

【図1】



K3900

*【図10】従来例において、各オリジナルタスクを起動する場合にリアルタイム処理が不可であることを示す図である。

【符号の説明】

RID1…リアルタイム情報処理装置、

10…CPU、

20…専用ハードウェア、

T0…オリジナルタスク、

Ts…サブタスク、

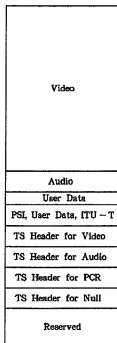
BPL…基本ボーリングループ、

N_{BPL}…基本ボーリングループの起動回数。

【図3】

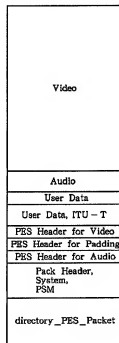
(1)

トランスポートストリーム
処理のためのメモリマップ



(2)

プログラムストリーム処理
のためのメモリマップ



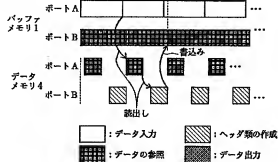
K3900

【図2】

(1) ソフトウェアによる制御方式

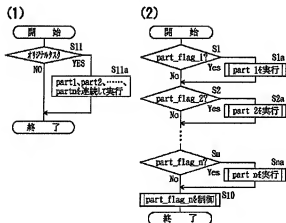


(2) 本発明

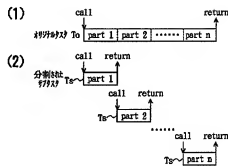


K3900

【図4】

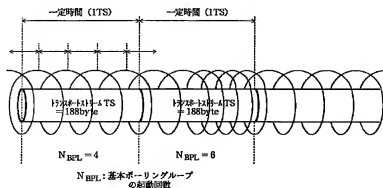


【図5】



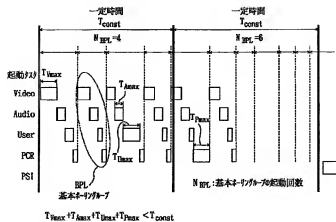
K3900

【図7】



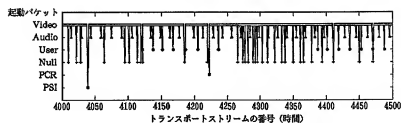
K3900

【図6】



K2900

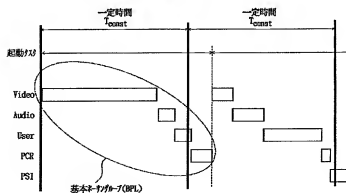
【図8】



K2900

【図10】

従来例における各タスクの起動状態 (リアルタイムの知能は不可)



K2900

【図9】

従来例におけるMPEG2のTSの多重化処理の動作

